Exercises for Hoare Logic

Jean Pichon-Pharabod

2018/2019

This exercise sheet is based on previous exercise sheets by Kasper Svendsen and by Mike Gordon. Mike Gordon’s exercise sheet also contains additional exercises: https://www.cl.cam.ac.uk/teaching/1516/HLog+ModC/MJCG-HL-Exercises.pdf.

Recommended exercises metatheory: 1, 22; practice: 2, 9, 35; specifications: 24, 25, 27; invariants: 12, 36, 37, 41; representation predicates: 47.

All the proof invariant exercises that do not involve separation logic can be formalised in Why3: http://why3.lri.fr/try/.

Exercise 1. Give a program $C$ such that the following partial correctness triple holds, or argue why such a $C$ cannot exist:

$$\{X = x \land Y = y \land x \neq y\} C \{x = y\}$$

Exercise 2. Show that the alternative assignment axiom

$$\{P\} \ X := E \ \{P[E/X]\}$$

is unsound by providing $P$ and $E$ such that

$$\neg(\models \{P\} \ X := E \ \{P[E/X]\})$$

Exercise 3 (Soundness of Floyd’s assignment axiom). Show that the alternative assignment axiom

$$x \notin FV(P)$$

$$\frac{}{\{P\} \ X := E \ \exists x. E[x/X] = X \land P[x/X]}$$

is sound.
Exercise 4 (Relative completeness of Floyd’s assignment axiom). Show that if we replace the assignment axiom by the following alternative assignment axiom

\[ x \notin FV(P) \]

\[ \{ P \} \ x := E \{ \exists x. E[x/X] = X \land P[x/X] \} \]

then the original assignment axiom is derivable.

Exercise 5. Show the soundness of the following rule:

\[ \vdash \{ P \} C \{ Q \} \quad \vdash \{ P \} C \{ R \} \]

\[ \vdash \{ P \} C \{ Q \land R \} \]

Exercise 6. Show the soundness of the following rule:

\[ \vdash \{ P \} C \{ R \} \quad \vdash \{ Q \} C \{ R \} \]

\[ \vdash \{ P \lor Q \} C \{ R \} \]

Exercise 7. Give a sound and relatively complete rule for a `repeat C until B` command (which is syntactic sugar for `C; while not B do C`).

Exercise 8. Prove that the following backwards reasoning sequenced assignment rule is derivable from the normal proof rules of Hoare logic:

\[ \{ P \} C \{ Q[E/X] \} \]

\[ \{ P \} C; X := E \{ Q \} \]

Exercise 9. Prove or give a counterexample for the following triple:

\[ \{ X = x \land Y = y \} \]

\[ X := X + Y; Y := X - Y; X := X - Y \]

\[ \{ Y = x \land X = y \} \]

Exercise 10. Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

\[ \{ X = x \land Y = y \land Y \geq 0 \} \]

\[ while \ Y > 0 \ do \ (X := X + 1; Y := Y - 1) \]

\[ \{ X = x + y \} \]
Exercise 11. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 12. Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

\[
\{ X = x \land Y = y \land Y \geq 0 \} \\
Z := 0; \\
A := 1; \\
\text{while } A \leq Y \text{ do (} Z := Z + X; A := A + 1 \) \\
\{ Z = x \times y \}
\]

Exercise 13. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 14. Recall that

\[
\vdash \forall x. \gcd(x, x) = x \\
\vdash \forall x, y. \gcd(x, y) = \gcd(y, x) \\
\vdash \forall x, y. x > y \Rightarrow \gcd(x, y) = \gcd(x - y, y)
\]

Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

\[
\{ X = x \land Y = y \land x > 0 \land y > 0 \} \\
\text{while } X \neq Y \text{ do (if } X > Y \text{ then } X := X - Y \text{ else } Y := Y - X \) \\
\{ X = Y \land X = \gcd(x, y) \}
\]

Exercise 15. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 16. Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

\[
\{ X = x \land Y = y \} \\
Z := 0; \\
\text{while not (} X = 0 \) do \\
\left( \begin{array}{l}
\text{if } X \text{ mod } 2 = 1 \text{ then } Z := Z + Y \text{ else skip}; \\
Y := Y \times 2; \\
X := X \text{ div } 2 \\
\end{array} \right) \\
\{ Z = x \times y \}
\]

Hint: \( X = (X \text{ div } 2 + X \text{ div } 2 + X \text{ mod } 2). \)
Exercise 17. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 18 (Fast exponentiation). Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

\[
\{ X = x \land N = n \land n \geq 0 \}
\]
\[
Z := 1;
\]
\[
\textbf{while } N > 0 \textbf{ do}
\]
\[
\begin{cases}
(\text{if } N \mod 2 = 1 \text{ then } Z := Z \times X \text{ else skip}); \\
N := N \div 2; \\
X := X \times X
\end{cases}
\]
\[
\{ Z = x^n \}
\]

Exercise 19. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 20 (Turing's large routine). Give a proof outline, and in particular loop invariants, for the following partial correctness triple:

\[
\{ N = n \land n \geq 0 \}
\]
\[
R := 0;
\]
\[
U := 1;
\]
\[
\textbf{while } R < N \textbf{ do}
\]
\[
\begin{cases}
S := 1; V := U; \\
\textbf{while } S \leq R \textbf{ do}
\begin{cases}
U := U + V; S := S + 1 \\
R := R + 1;
\end{cases}
\end{cases}
\]
\[
\{ U = \text{fact}(n) \}
\]

Exercise 21. Give variants to obtain a total correctness triple for the same pre- and postcondition and command.

Exercise 22. Prove soundness of the separation logic heap assignment rule by proving that

\[
|= \{ E_1 \mapsto t \} [E_1] := E_2 \{ E_1 \mapsto E_2 \}
\]

Exercise 23. Formalise and prove that if \( X \mapsto t_1 \land Y \mapsto t_2 \), then \( X \) and \( Y \) alias, and \( t_1 \) and \( t_2 \) are equal.
Exercise 24. Give a triple specifying that a command $C$ orders the values of $X$ and $Y$, so that the smaller value ends in $X$, and the greater value in $Y$.

Exercise 25. Give a triple specifying that a command $C$ computes into $Z$ the sum of $X$ and $Y$ if $R$ is 0, and their product otherwise.

Exercise 26. Give a triple specifying that a command $C$ sorts a list starting at $X$.

Exercise 27. Give a triple specifying that a command $C$ concatenates a list starting at $X$ with itself.

Exercise 28. Give a triple specifying that a command $C$ appends the value of $V$ to the start of a list starting at $X$ if $R$ is 0, and to the end of a list at $Y$ otherwise.

Exercise 29. Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

$\{N = n \land n \geq 0 \land X = 0 \land Y = 0\}$

$\textbf{while } X < N \textbf{ do } (X := X + 1; Y := Y + X)$

$\{Y = \sum_{i=1}^{n} i\}$

Exercise 30. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 31 (Euclid’s algorithm). Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

$\{X = x \land Y = y\}$

$R := X;$

$Q := 0;$

$\textbf{while } Y \leq R \textbf{ do }$

$(R := R - Y; Q := Q + 1)$

$\{x = R + y \times Q \land R < y\}$

Exercise 32. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).
Exercise 33 (Divisibility by 13). Give a proof outline, and in particular a loop invariant, for the following partial correctness triple:

\[
\begin{align*}
\{ & X = x \land X \geq 0 \} \\
\text{while } & X \geq 52 \text{ do} \\
& X := (X \div 10) + 4 \times (X \mod 10); \\
& \text{if } X = 0 \text{ or } X = 13 \text{ or } X = 26 \text{ or } X = 39 \text{ then } Y := 1 \text{ else } Y := 0 \\
\{ & Y = 1 \iff x \mod 13 = 0 \}
\end{align*}
\]

Exercise 34. Give a variant to obtain a total correctness triple (you might need to strengthen the precondition and the invariant).

Exercise 35. Give a proof outline for the following separation logic partial correctness triple:

\[
\begin{align*}
\{ & \text{list}(X, \alpha) \} \\
\text{if } & X = \text{null} \text{ then } Y := \text{null} \\
\text{else } & (E := [X]; P := [X+1]; Y := \text{alloc}(E, P); \text{dispose}(X); \text{dispose}(X+1)) \\
\{ & \text{list}(Y, \alpha) \}
\end{align*}
\]

Exercise 36. Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

\[
\begin{align*}
\{ & \text{list}(X, \alpha) \} \\
& Y := \text{null}; \\
\text{while } & X \neq \text{null} \text{ do} \\
& (Z := [X+1]; [X+1] := Y; Y := X; X := Z) \\
\{ & \text{list}(Y, \text{rev}(\alpha)) \}
\end{align*}
\]

where \( \text{rev} \) is mathematical list reversal, so that

\[
\begin{align*}
\text{rev}([]) &= [] \\
\text{rev}([h]) &= [h] \\
\text{rev}(\alpha ++ \beta) &= \text{rev}(\beta) ++ \text{rev}(\alpha)
\end{align*}
\]

Exercise 37. Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

\[
\begin{align*}
\{ & \text{list}(X, \alpha) \} \\
& N := 0; \\
& Y := X; \\
\text{while } & Y \neq \text{null} \text{ do} \\
& (N := N + 1; Y := [Y + 1]) \\
\{ & \text{list}(X, \alpha) \land N = \text{length}(\alpha) \}
\end{align*}
\]
Exercise 38. Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

\[
\{ N = n \land \text{emp} \}
\]

if \( N \leq 0 \) then \( X := \text{null} \)

\[
\begin{align*}
X &:= \text{alloc}(0, \text{null}); \\
P &:= X;
\end{align*}
\]

else

\[
I := 1;
\]

while \( I < N \) do

\[
\begin{align*}
(Q &:= \text{alloc}(I, \text{null}); [P + 1] := Q; P := Q; I := I + 1) \\
\end{align*}
\]

\[
\{ \text{list}(X, 0 :: \ldots :: n - 1 :: []) \land N = n \}
\]

Exercise 39. Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

\[
\{ \text{list}(X, \alpha) \}
\]

\[
Y := \text{alloc}(0, \text{null}); Y' := Y; \\
Z := \text{alloc}(0, \text{null}); Z' := Z;
\]

while \( X \neq \text{null} \) do

\[
\begin{align*}
[Y' + 1] &:= X; Y' := X; X := [X + 1]; \\
\text{if } X \neq \text{null} \text{ then } ([Z' + 1] := X; Z' := X; X := [X + 1]) \text{ else skip}
\end{align*}
\]

\[
\begin{align*}
[Y' + 1] &:= \text{null}; \\
[Z' + 1] &:= \text{null}; \\
U &:= [Y + 1]; \text{dispose}(Y); \text{dispose}(Y + 1); Y := U; \\
U &:= [Z + 1]; \text{dispose}(Z); \text{dispose}(Z + 1); Y := U;
\end{align*}
\]

\[
\{ \exists \alpha_1, \alpha_2. \text{length}(\alpha) = \text{length}(\alpha_1) + \text{length}(\alpha_2) \land (\text{list}(Y, \alpha_1) \ast \text{list}(Z, \alpha_2)) \}
\]

Exercise 40. Give a proof outline, and in particular a loop invariant, for the same separation logic partial correctness triple, but with the following postcondition:

\[
\{ \exists \alpha_1, \alpha_2. \text{shuffle}(\alpha, \alpha_1, \alpha_2) \land (\text{list}(Y, \alpha_1) \ast \text{list}(Z, \alpha_2)) \}
\]

where

\[
\text{shuffle}([], [], []) \overset{\text{def}}{=} \top
\]

\[
\text{shuffle}(x :: \alpha, \beta, \gamma) \overset{\text{def}}{=} (\exists \beta'. \beta = x :: \beta' \land \text{shuffle}(\alpha, \beta', \gamma)) \lor (\exists \gamma'. \gamma = x :: \gamma' \land \text{shuffle}(\alpha, \beta, \gamma'))
\]
Exercise 41. Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

\[
\{\text{list}(X, \alpha) \land \text{sorted}(\alpha) \land Y = y\}
\]

if \(X = \text{null}\) then \(X := \text{alloc}(Y, \text{null})\)

else

\[
\begin{cases}
P := X; E := [P]; \\
\text{if } Y \leq E \text{ then } X := \text{alloc}(Y, X) \\
\text{else}
\begin{cases}
Q := P; \\
\text{while } E < Y \text{ and } P \neq \text{null} \text{ do}
\begin{cases}
Q := P; P := [P + 1]; E := [P]; \\
R := \text{alloc}(Y, P) \\
[Q + 1] := R
\end{cases}
\end{cases}
\end{cases}
\]

\[
\exists \alpha_1, \alpha_2. \ (\forall i. 0 \leq i < \text{length}(\alpha_1) \Rightarrow \alpha_1[i] < y) \land \\
(\forall i. 0 \leq i < \text{length}(\alpha_2) \Rightarrow y \leq \alpha_2[i]) \land \\
\text{list}(X, \alpha_1 \oplus [y] \oplus \alpha_2)
\]

Exercise 42. Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

\[
\{\text{list}(X, \alpha)\}
\]

if \(X = \text{null}\) then \(Y := \text{null}\)

else

\[
\begin{cases}
P := X; E := [P]; Y := \text{alloc}(E, \text{null}); Q := Y; P := [X + 1]; \\
\text{while } P \neq \text{null} \text{ do}
\begin{cases}
E := [P]; Q_2 := \text{alloc}(E, \text{null}); [Q + 1] := Q_2; Q := Q_2; P := [P + 1]
\end{cases}
\end{cases}
\]

\[
\{\text{list}(X, \alpha) \ast \text{list}(Y, \alpha)\}
\]

Exercise 43 (Index search). Give a proof outline, and in particular a loop
invariant, for the following separation logic partial correctness triple:

\[
\{ X = x \land x \in_{\text{list}} \alpha \land \text{list}(Y, \alpha) \} \\
I := 0; Z := Y; S := 0; \\
\text{while } S = 0 \text{ do} \\
\left( \begin{array}{l}
E := \llbracket Z \rrbracket; \\
\text{if } E = X \text{ then} \\
S := 1 \\
\text{else} \\
(Z := \llbracket Z + 1 \rrbracket; I := I + 1)
\end{array} \right) \\
\{ \alpha[I] = x \land \text{list}(Y, \alpha) \}
\]

where \( \in_{\text{list}} \) is list membership:

\[
\begin{align*}
\& x \in_{\text{list}} \emptyset \overset{\text{def}}{=} \bot \\
\& x \in_{\text{list}} (y :: \beta) \overset{\text{def}}{=} (x = y) \lor (x \in_{\text{list}} \beta)
\end{align*}
\]

**Exercise 44** (Prefix testing). Give a proof outline, and in particular a loop invariant, for the following separation logic partial correctness triple:

\[
\{ \text{list}(X, \alpha) \ast \text{list}(Y, \beta) \} \\
P := X; Q := Y; S := 1; \\
\text{while } S = 1 \text{ and } P \neq \text{null} \text{ and } Q \neq \text{null} \text{ do} \\
\left( \begin{array}{l}
E := \llbracket P \rrbracket; F := \llbracket Q \rrbracket; \\
\text{if } E = F \text{ then} \\
(P := \llbracket P + 1 \rrbracket; Q := \llbracket Q + 1 \rrbracket) \\
\text{else} \\
S := 0 \\
\{ \text{list}(X, \alpha) \ast \text{list}(Y, \beta) \land (S = 0 \Leftrightarrow \neg(\alpha \sqsubseteq \beta \lor \beta \sqsubseteq \alpha)) \}
\end{array} \right)
\]

where \( \sqsubseteq \) is prefix relation:

\[
\begin{align*}
\& \emptyset \sqsubseteq \beta \overset{\text{def}}{=} \top \\
\& h :: \alpha \sqsubseteq \beta \overset{\text{def}}{=} \exists \gamma. \beta = h :: \gamma \land \alpha \sqsubseteq \gamma
\end{align*}
\]

**Exercise 45** (Substring testing). Give a proof outline, and in particular a
loop invariant, for the following separation logic partial correctness triple:

\[
\begin{align*}
&\{\text{list}(X, \alpha) \ast \text{list}(Y, \beta)\} \\
&S := 1; P := X; Q := Y; \\
&\textbf{while } (S = 1 \text{ and } P \neq \text{null}) \textbf{ do} \\
&\quad \begin{cases} \\
&\quad \quad \text{if } Q = \text{null} \text{ then } S := 0 \\
&\quad \quad \text{else} \\
&\quad \quad \quad \begin{cases} \\
&\quad \quad \quad \quad E := [P]; F := [Q]; \\
&\quad \quad \quad \quad \text{if } E = F \text{ then } P := [P + 1] \\
&\quad \quad \quad \quad \text{else skip;} \\
&\quad \quad \quad \quad Q := [Q + 1] \\
&\quad \quad \end{cases} \\
&\quad \quad \{ (S = 0 \iff (\alpha \sqsubseteq \beta) \land (\text{list}(X, \alpha) \ast \text{list}(Y, \beta))) \} \\
&\quad \end{cases}
\end{align*}
\]

where \(\sqsubseteq\) is the (not-necessarily-contiguous) substring relation:

\[
\begin{align*}
&[] \sqsubseteq \beta \overset{\text{def}}{=} T \\
&h :: \alpha \sqsubseteq \beta \overset{\text{def}}{=} (\exists \gamma. \beta = h :: \gamma \land \alpha \sqsubseteq \gamma) \lor (\exists i, \gamma. \beta = i :: \gamma \land h :: \alpha \sqsubseteq \gamma)
\end{align*}
\]

Exercise 46 (Bubble sort). Give a proof outline, and in particular loop invariants, for the following separation logic partial correctness triple:

\[
\begin{align*}
&\{\text{list}(X, \alpha)\} \\
&D := 0; \\
&\textbf{while } D = 0 \textbf{ do} \\
&\quad \begin{cases} \\
&\quad \quad S := 1; P := X; \\
&\quad \quad \textbf{while } P \neq \text{null} \textbf{ do} \\
&\quad \quad \quad \begin{cases} \\
&\quad \quad \quad \quad Q := [P + 1]; \\
&\quad \quad \quad \quad \text{if } Q \neq \text{null} \text{ then} \\
&\quad \quad \quad \quad \quad \begin{cases} \\
&\quad \quad \quad \quad \quad \quad E := [P]; F := [Q]; \\
&\quad \quad \quad \quad \quad \quad \text{if } E \leq F \text{ then} \\
&\quad \quad \quad \quad \quad \quad \quad P := Q \\
&\quad \quad \quad \quad \quad \quad \text{else} \\
&\quad \quad \quad \quad \quad \quad \quad (S := 0; [P] := F; [Q] := E) \\
&\quad \quad \quad \quad \quad \quad \text{else skip} \\
&\quad \quad \quad \quad \quad \quad \quad \text{if } S = 1 \text{ then } D := 1 \text{ else skip} \\
&\quad \quad \quad \quad \end{cases} \\
&\quad \quad \quad \quad \text{else} \\
&\quad \quad \quad \quad \quad \quad (S := 0; [P] := F; [Q] := E) \\
&\quad \quad \quad \quad \quad \quad \text{else skip} \\
&\quad \quad \end{cases} \\
&\quad \end{cases} \\
&\{ \exists \beta. \text{sorted}^\beta \land \text{permutation}(\alpha, \beta) \land \text{list}(X, \beta) \}
\end{align*}
\]

Exercise 47. Give a representation predicate \(btree(t, \tau)\) for binary trees, given a mathematical representation \(\tau ::= \text{Leaf} \mid \text{Node } n \tau_1 \tau_2\), where \(n\) is an integer.
Exercise 48. Give a representation predicate $clist(t, \alpha)$ for circular lists.

Exercise 49. Give a representation predicate $list'(t, \alpha)$ for doubly-linked lists.

Exercise 50. Give a representation predicate $array(t, \alpha)$ for arrays starting at location $t$, the contents of which is represented by the mathematical list $\alpha$. 